Cryptographic Practices

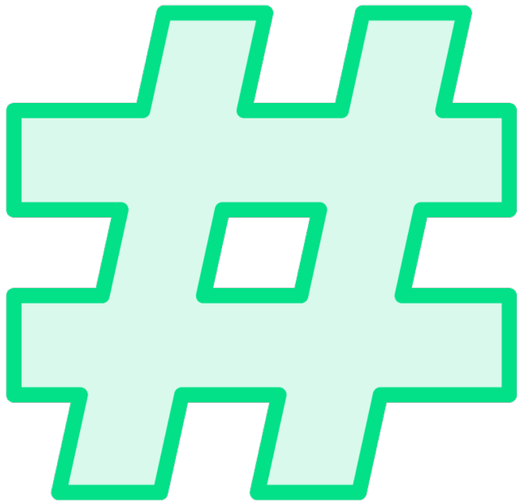
Paul Mooney
Chief Software Architect, Microsoft MVP

@daishisystems | www.insidethecpu.com

Overview

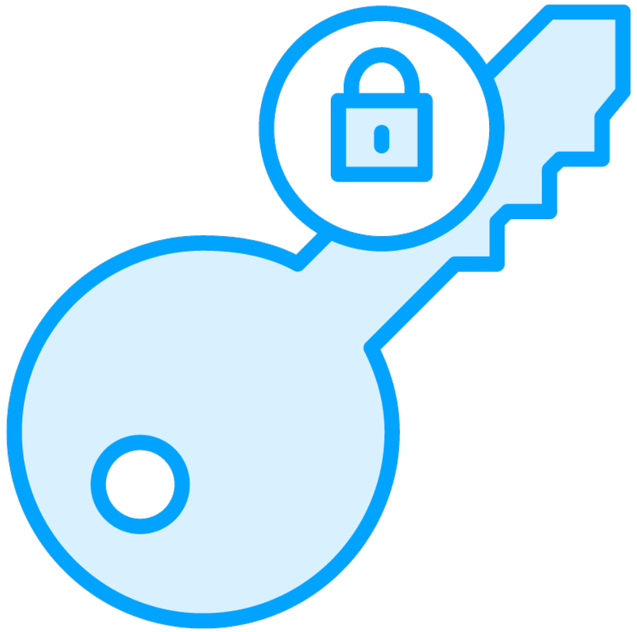
Cryptographic Practices

- Cryptographic practices in Go
- Hashing and encrypting differ
- Distinct characteristics, different objectives
- Unravelling the mysteries of hashing
- Unravelling the mysteries of encryption
- Effective web security with cryptography



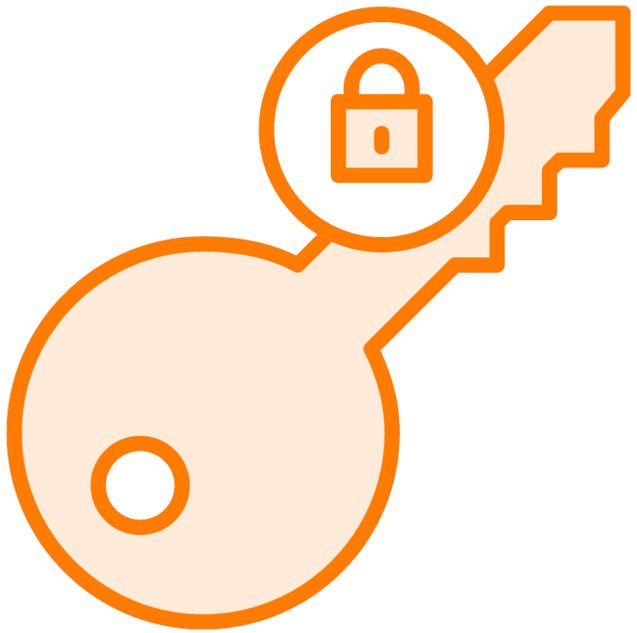
Hashing

- Hash: generated from source data
- Fixed length, varying with input
- Well-designed algorithms prevent reverse-engineering
- BLAKE2: strongest and flexible algorithm
- Go language supports BLAKE2b and BLAKE2s
- SHA-256 as reliable alternative
- Slow hashing defends against attacks
- Minimum of 10,000 iterations recommended
- Hashing for verifying data integrity



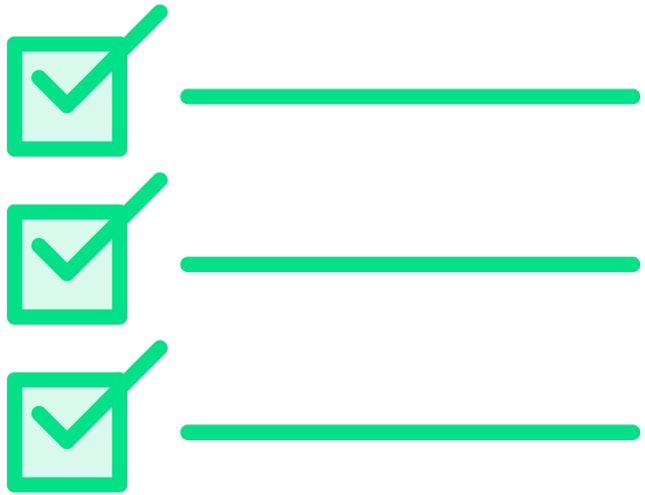
Encryption

- Encryption transforms data with a key
- Encryption is reversible, unlike hashing
- Safeguards data and ensures authorized access
- Used for secure communication and storage
- Example: HTTPS protocol
- Symmetric key encryption: Advanced Encryption Standard (AES)
- AES with modes like GCM provides authentication
- Public key cryptography involves key pairs
- Go supports modern symmetric encryption algorithms



NaCL

- Go programming and NaCl: Trusted cryptography library
- nacl/box: Authenticated, encrypted messaging with public key cryptography
- nacl/secretbox: Authenticated, encrypted messaging with symmetric cryptography
- Utilize NaCl abstractions for secure messaging scenarios
- Higher-level, more secure cryptographic operations than AES
- Achieve secure and authenticated communication in Go applications



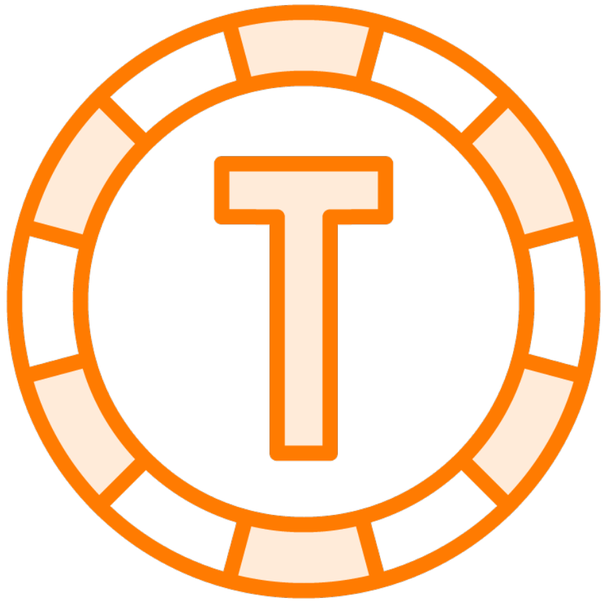
Best Practices

- Establish and adhere to key management policies
- Protect "master secrets" from unauthorized access
- Avoid hardcoding cryptographic keys
- Go's crypto package offers common cryptographic constants
- Specific algorithms have dedicated packages
- Use algorithms from the x/crypto repository
- Numerous contemporary cryptographic algorithm implementations available
- Utilize libraries from reputable sources
- Enhance security and fortify Go applications

Summary

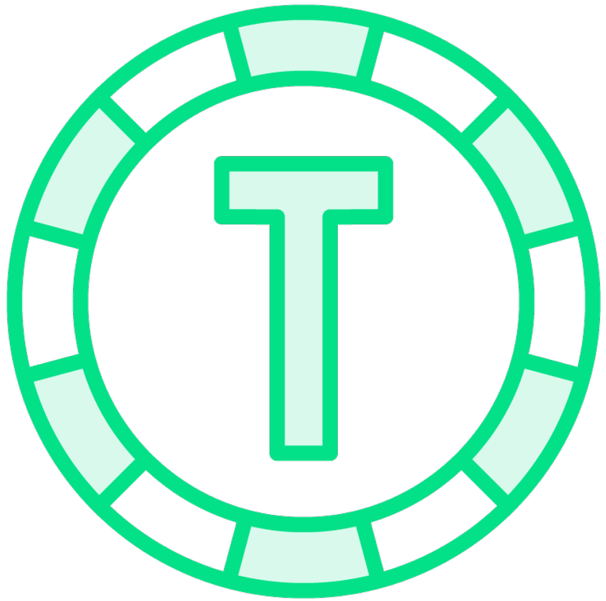
Cryptographic Practices

- Emphasize distinction between hashing and encryption
- Highlight fixed-length output and integrity verification of hashing
- Encryption: key-based cipher text transformation
- Advanced Encryption Standard (AES) as standard symmetric encryption algorithm
- Galois Counter Mode (GCM) as authenticated cipher mode
- Public key cryptography and modern symmetric encryption algorithms



JWT Benefits

- Stateless session
- Improved scalability
- Suitable for distributed architectures



Session Generation

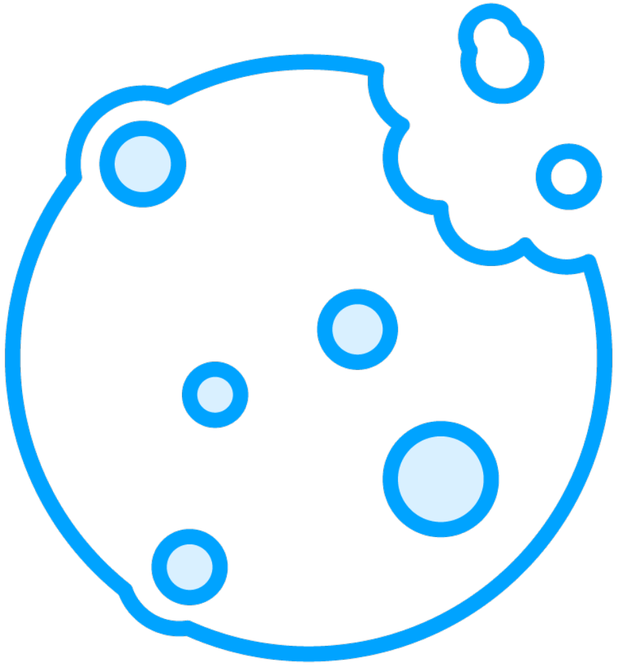
- JWT configuration
- Claims and attributes
- Token integrity

Session Management

- Secure session identifiers
- Essential cookie attributes

Secure Session Identifiers

- Sufficient randomness
- Unpredictable
- Difficult to guess

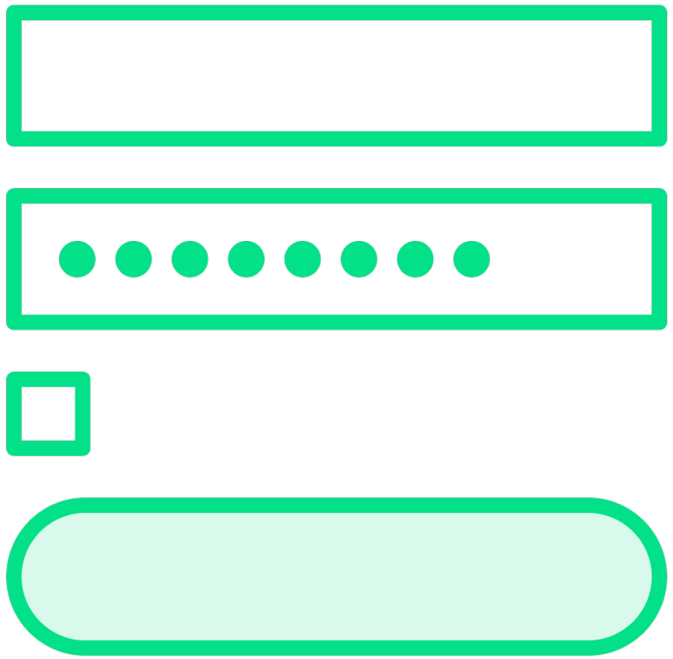


Cookie Attribute Configuration

- Client session identifier store
- Domain
- Path
- Expires
- HttpOnly
- Secure

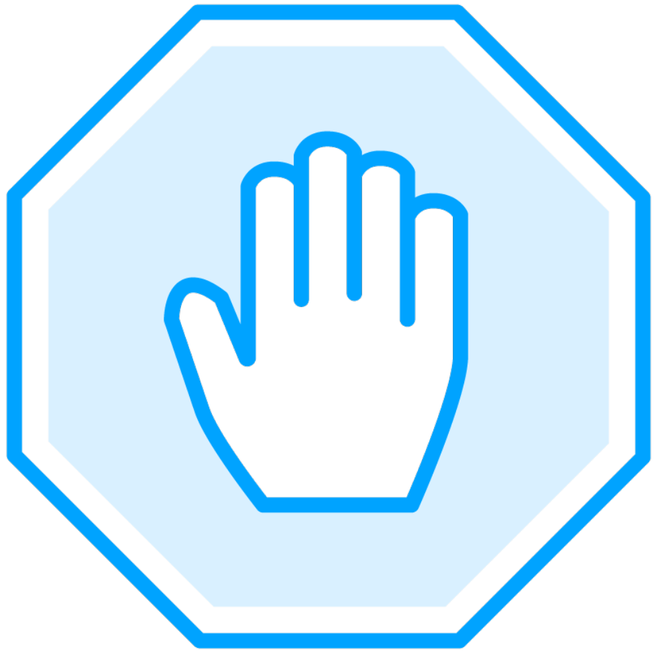
Session Integrity

- Generate new session on sign-in
- Don’t reuse sessions
- Mitigates session-related vulnerabilities
  - Session fixation
  - Session hijacking
- Periodic session termination



Concurrent Login Prevention

- Logged-in user database
- Prevents unauthorized access
- Transmit in HTTP cookie header

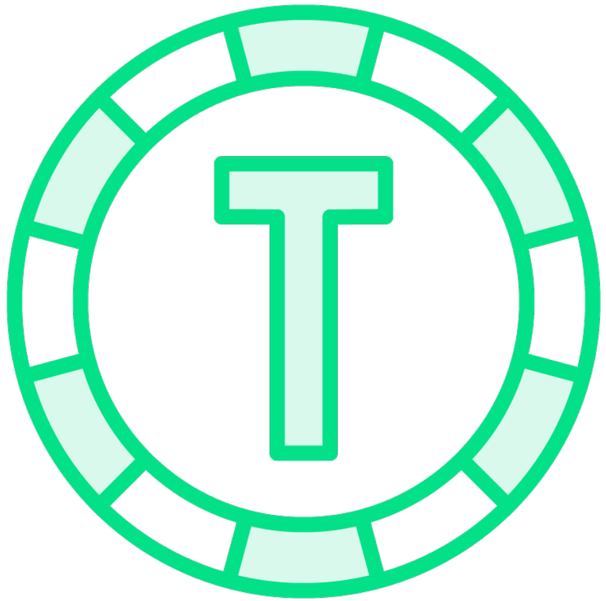


Access Controls

- Restrict session data
- Only accessible to authenticated user
- User isolation
- Access restriction
- Maintain session privacy
- Prevent data leakage

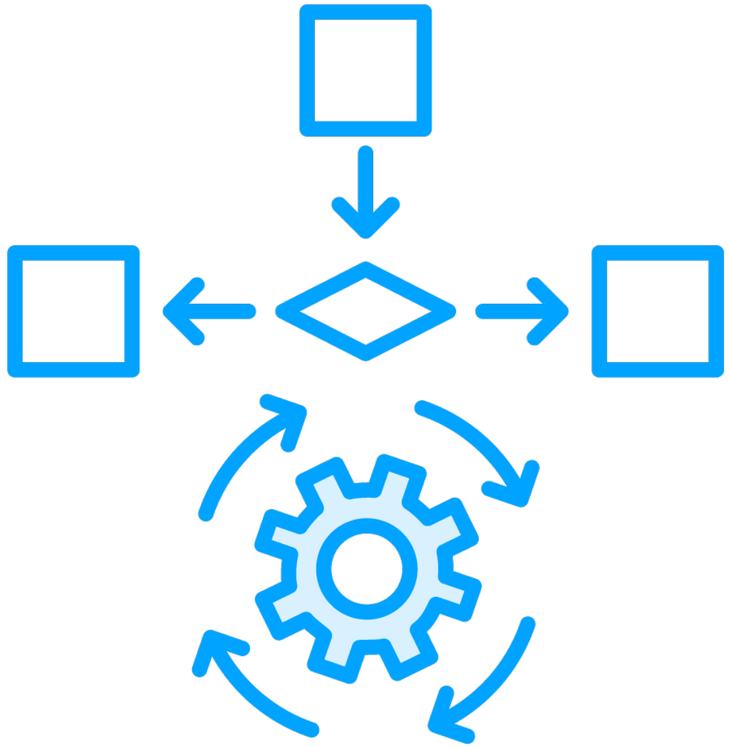
Man-in-the-middle (MITM) Attacks

- Intercept and hijack user sessions
- Use HTTPS
- Secure channel for communication



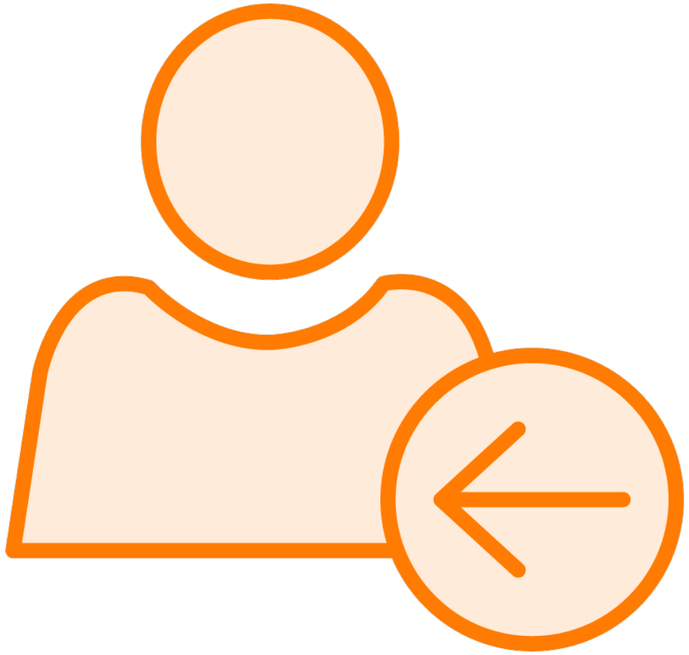
Tokens and Logout

- Per-request tokens
- Logout functionality



Critical Operations

- Per-request basis
- Each operation has a unique token
- Limited validity



Logout Functionality

- Session termination
- Disconnect from authenticated pages
- Connection termination

Demo

Secure Session Management

Summary

Session Management

- Manage sessions securely
- Maintaining user state
- Session initialization
- Hijacking and fixation attacks
- Logout functionality and timeouts
- Transmission over secure networks
- JSON web tokens (JWTs)
- Secure session identifiers
- Session cookies attributes

Communicating Authentication Data

Single-factor authentication

- Single factor for identification
- Easy to implement
- Least secure

Multi-factor authentication

- Multiple factors of identification
- More secure than single-factor
- More complex to implement

Biometric authentication

- Physical or behavioral characteristic
- Most secure type of authentication
- Expensive and difficult to implement

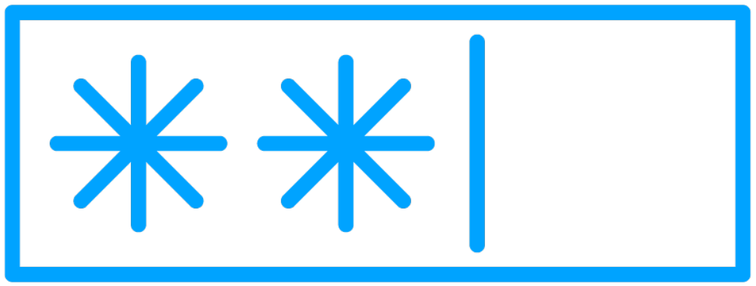
Common Authentication Protocols

- OAuth and OpenID Connect
- Limited access to resources
- 3rd party authentication
- Complex implementation
- Standardized way to authenticate users
- Use secure transport protocols
- Validate all input data
- Principle of least privilege
- Stay up to date

Secure Communication Protocols

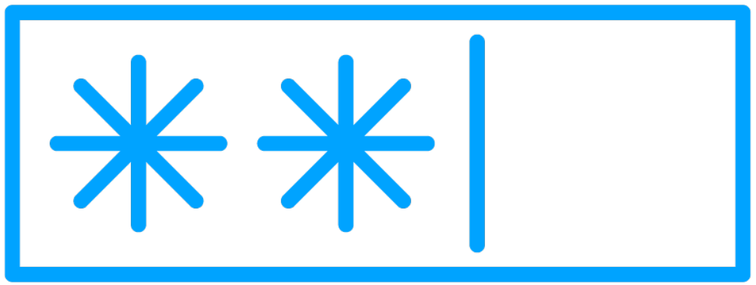
- Transport Layer Security (TLS)
- Secure Sockets Layer (SSL)
- Properly configuring certificates
- Disable outdated cryptographic ciphers
- Apply the latest security patches
- Use HTTP POST

Validating & Storing Authentication Data



Password Security Best Practices

- Hashing
- Salting
- Encryption



Password Policies

- Password length
- Password complexity
- Expiration date

Password Storage Options

Hashing

Fixed length

Encrypted

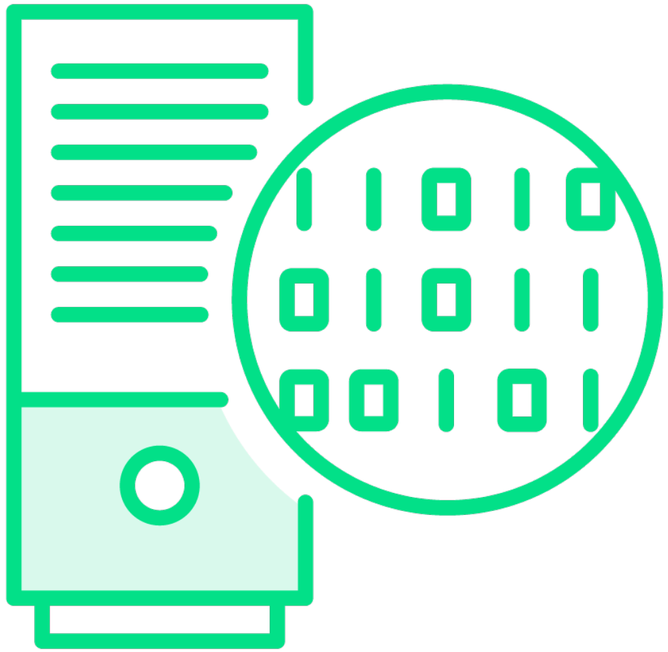
One-way

Salting

Random characters

Precomputed hash tables

Unique per password

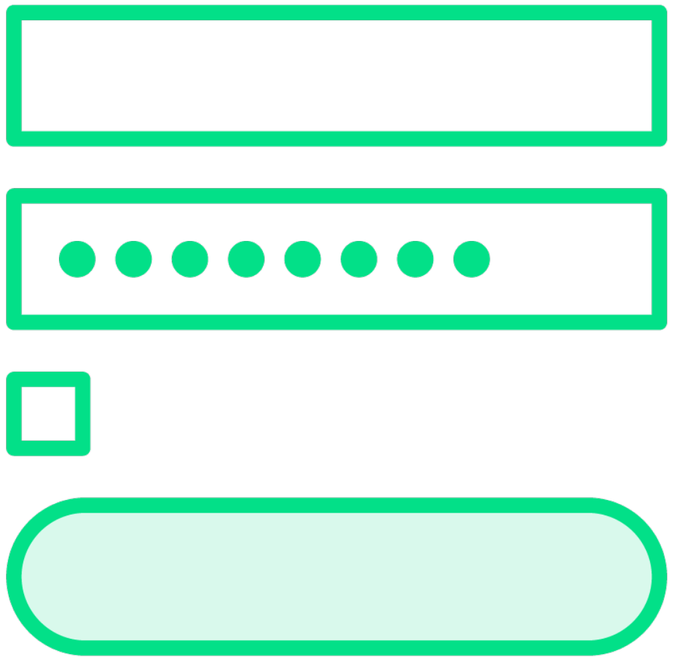


Supported Hashing Algorithms

- Bcrypt
- Scrypt
- Salt value per password

Demo

Single-factor Authentication



Password Policies

- Password complexity
- Hashing
- Salting
- Expiration
- Account lockout
- Policies versus convenience

Password Reset

- Requesting a reset
- Token storage
- Reset link generation
- Reset password page
- Password update
- Token expiration
- Token revocation
- Logging
- Error handling

Demo

Multi-factor Authentication

Summary

Authentication & Password Management

- Secure implementation guidelines
- Authentication types
- Common protocols
- Secure communication protocols
- Password security best practices
- SFA and MFA demos